write a SQL query to report the type of each node in a tree each node can be one of three types Leaf root or inner for those of you who aren't already familiar with exponent exponent helps you get your dream Tech Career with our online courses expert coaching peer-to-peer mock interviewing platform and interview question database check it out at try exponent.com [Music] cool yeah so um we have it here is tree node table Community table and right here we see basically we have two columns ID and PID and yeah so from there I would like to ask a few clarifying questions maybe just one main clarifying question that's just basically about the inner so an inner could just basically be not the root but a parent but also a child correct uh yes okay cool um and basically what I'm trying to do is identify each ID based on the definition yes cool okay if you give me a second I'll maybe think about how I'm gonna attack this a few techniques in mind um but then yeah then I'll get started okay sounds good cool so yeah I'm thinking of just using a simple case statement um to Define each of these um IDs respectively so we can do a case when um PID like right here is no then root right and then and just to clarify here PID means parent ID right yeah yeah parent ID to yeah the node basically um so yeah that makes sense and then when um I think I'm now going to Define an inner so when the PID is and making a sub query here select ID and the ID column from the tree note table um and when the ID is in the tree note table of the PID column s then it would be an inner right does that make sense yes cool um in and let's check my syntax here okay so they're both okay and then uh else it's a leaf um and then I want to end this and maybe call this column no types um so from here I'd just to be safe I'm going to order it by the ID um and I just want to see if it'll work so oops cool so um this does seem to be correct so the ID would be the number one ID would be the root because it was no um and three four and five would be leaves because they all three of them are parents um and the inner would be number two um as it's apparent and a leaf at the same time making it an inner um so yeah um okay uh and can you also describe to me in words like uh like which which one of these notes was like the child of which node just so we can't get a better picture of what the tree actually looks like yeah so um the root being one the number two node um was a child to the number one node and the number the number three node was also a child to the number one node but the number two node had two other children or Leafs basically um and that was four four and five okay yeah perfect um so let's check a real world scenario and so let's say that this tree actually corresponds to an employee directory where um so each node corresponds to one employee and then the children node of a node are basically the reports of that particular employee and the IDS are basically the employee IDs so let's assume that like you have there's one table that describes like the employee directory right but the actual employee information like the names of the employees um and their um like salaries or types of jobs um types of positions are actually stored in a separate table uh that also has the employee ID as a key so how would you then um uh basically let's say that you're given like a specific um employee how would you write like the query that determines uh all of the contractors um names that are the reports of a specific employee cool yeah got it let me just really quickly comment this out cool so um I think it would be cool to show the employee table so I have a better understanding um of what I'm looking for and just to be clear you wanted the title slash so just the full names of all of the contractors that are the reports of a particular employee cool um uh some output would be first as well as titles Exposition being contractor um okay cool cool and also um just because like our employee directory tells us like specifically the relationships I'd like like let's say I give you a specific employee ID I want you to find only the ones uh the contractors who are the reports of that employee okay who are the who are the reports of that employee okay cool um so from here just looking at the employee directory um you can see here there's like two contractors cool um okay got it so okay so what I understand is so so you want the and the okay so let's say that there's these two contractors um you want their full names basically uh yeah so long as they are um the reports of a specific let's say I think it's employee number two if I remember correctly yeah okay cool um one way we could find this is through joining on um so listening or yeah and the table T and then e would be employee ID correct yeah you know employee underscore ID which is equal to uh T dot ID and what we're looking for is basically the idea of like Eid I'm sorry T dot ID um e DOT first underscore name and then t or sorry e Dot sticker squaring um and so they have to be yeah okay um correct like um okay so check my syntax table even the first and last name um joining on the employee ID from the employee table right and then T from the tree note table yeah also their manager is employee number two right um and then you don't manager ID um yeah so here we have the respective ID it's Jon Snow and Patrick Star and they're both under contractor we can even just put Title Here to make that known maybe right here title like that yeah so they're both contractors Jon Snow and Patrick Starr and their manager's number has a manager ID number two oh okay perfect so a couple of quick follow-up questions on this um so I've noticed then that um the original table the tree table you have like parent ID right which represents the manager ID and then in the employee table you also have the manager ID field so I'm wondering do you still need both of these columns um good question there's a PID technically no they're both representative of the same thing just both within different columns between different tables yeah and since you're like doing this join query anyway so you can easily just pull the manager ID from the P ID column and uh um tree table okay exactly great so I think this is a great place to pause um I thought you did a fantastic job in this interview and I'd love to hear from you uh what do you think went well and what do you think you could improve upon thanks yeah I think one thing right off the bat that I could have improved upon was just optimizing my query so maybe using this first one finding the answer and then meaning defining each node as a root inner leaf and then using it as a sub query because at the end of the day we're basically just pulling from this ID is the primary key and like we talked about last time I could have just added the PID as well and we could have joined from either or um and then uses a sub query um and pulled that tid right here um or even connect that manager ID to that and it would just gone through once basically without even having to use a join so yeah yeah okay perfect um and so I'm also wondering uh is there a way that you can maybe add an index to one of the tables to make this query even faster right um in terms of indexing in the where Clause here I could have also just done an equal sign taking out this wild card to basically only tell the query or the MySQL machine just to look for the title contractor within our employee table which would have been quicker yeah so then which column are you doing the index on um it would be in the title column okay perfect great thanks so much I like similar thoughts actually um I really like that at the beginning of the interview you explain first like what are the in words like what are the characteristics that make an inner note so that it has to have parents and children because then a lot more clear before you dived into coding um and yeah I really like that you also um explain thoroughly like what exactly the join does um and then you like damage showed in the actual table like which columns that you're pulling out um I think yeah so aside from what we had just talked about like optimizing the query it can also be helpful sometimes to just explain like uh if you have a column name like PID like what does that mean exactly and um there are different types of joints too right so here um the default is like an inner join right so you did an energy um and sometimes it can be helpful to explain like why did you choose an inner joint instead of like left or right for example so yeah I totally agree those little syntactical nuances are really important especially when you're pulling from a bunch of different tables yeah just to show off like your knowledge of SQL you know yeah okay great thanks so much for being here with us today Christopher um I think we've learned a lot from you and that was a really cool question to get to see you answer okay thanks everybody for watching thanks so much for watching don't forget to hit the like And subscribe buttons below to let us know that this video is valuable for you and of course check out hundreds more videos just like this at try exponent.com thanks for watching and good luck on your upcoming interview foreign [Music]